

**AMENDMENTS TO THE CLAIMS**

Claims 1-5, 7-10, 15-19, 21-24, 29-43 and 48-66 were pending at the time of the Action.

No claims are canceled in this Response.

Claims 1, 5, 7-9, 15, 19, 21-23, 31, 36, 41, 50, 55, 60 and 63 are amended in this Response.

Claims 1, 7-9, 15, 21-23, 63 and 65 are independent claims.

Claims 1-5, 7-10, 15-19, 21-24, 29-43 and 48-66 are pending in the present Application.

Claim 1 (currently amended): A method for creating a binary tree data structure, the data structure embodied in a computer-readable medium, from an ordered list of at least four elements, each element having an associated value in the list, comprising:

determining whether the list has an even or odd number of elements;

separating the list into left side groupings and right side groupings based on whether the list has an even or odd number of elements, the groupings being separated by a parent node defined by a median of the list, wherein the median is a left element of two middle values of the list when the list has an even number of elements, or the median is a middle value element of the list when the list has an odd number of elements;

creating left side descendent nodes of the binary tree by successively finding a median of each left side grouping and linking each found median to the previous median, wherein a median of a first left side grouping is linked to the parent node;

creating right side descendent nodes of the binary tree by successively finding a median of each right side grouping and linking each found median to the

1 previous median, wherein a median of a first right side grouping is linked to the  
2 parent node;

3 wherein when a grouping has an even number of elements, the median is a  
4 left element of two middle values of the grouping;

5 wherein when a grouping has an odd number of number of elements, the  
6 median is a middle value element of the grouping; and

7 wherein the elements of the lists include logged events.

8 Claim 2 (original): A computer-readable medium having stored thereon  
9 computer-executable instructions for performing the method of claim 1.

10 Claim 3 (previously amended): The method of claim 1, wherein each  
11 element in the list includes a pointer to a corresponding node of a plurality of  
12 nodes in a partially assembled binary tree, wherein each node has a left child  
13 pointer, and wherein creating the left side nodes further comprises assigning a  
14 value to the left child pointer of at least one of the nodes.

15 Claim 4 (previously amended): The method of claim 1, wherein each  
16 element in the list includes a pointer to a corresponding node of a plurality of  
17 nodes in a partially assembled binary tree, wherein each node has a right child  
18 pointer, and wherein creating the right side nodes further comprises assigning a  
19 value to the right child pointer of at least one of the nodes.

20 Claim 5 (currently amended): The method of claim 1, wherein creating the  
21 left side descendent nodes comprises ~~inserting~~ linking the left side descendent  
22 nodes ~~into to~~ a partially assembled version of the binary tree, wherein creating the  
23 right side descendent nodes comprises ~~creating~~ linking the right side descendent  
24 nodes ~~into to~~ the partially assembled version of the binary tree, and wherein the  
25 list is a linked list that acts as a wrapper around the partially assembled version of

the binary tree.

Claim 6 (canceled).

Claim 7 (currently amended): A method for creating a binary tree data structure, the data structure embodied in a computer-readable medium, from an ordered list of at least four elements, each element having an associated value in the list, comprising:

determining whether the list has an even or odd number of elements;

separating the list into left side groupings and right side groupings based on whether the list has an even or odd number of elements, the groupings being groupings separated by a parent node defined by a median of the list, wherein the median is a left element of two middle values of the list when the list has an even number of elements, or the median is a middle value element of the list when list has an odd number of elements;

creating left side descendent nodes of the binary tree by successively finding a median of each left side grouping and linking each found median to the previous median, wherein a median of a first left side grouping is linked to the parent node;

creating right side descendent nodes of the binary tree by successively finding a median of each right side grouping and linking each found median to the previous median, wherein a median of a first right side grouping is linked to the parent node;

wherein when a grouping has an even number of elements, the median is a left element of two middle values of the grouping;

wherein when a grouping has an odd number of number of elements, the median is a middle value element of the grouping; and

wherein the elements of the list include data representing number of times one or more threads of execution have passed through one or more code modules.

1 Claim 8 (currently amended): A method for creating a binary tree data  
2 structure, the data structure embodied in a computer-readable medium, from an  
3 ordered list of at least four elements, each element having an associated value in  
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;  
6 separating the list into left side groupings and right side groupings based on  
7 whether the list has an even or odd number of elements, the groupings being  
8 separated by a parent node defined by a median of the list, wherein the median is a  
9 left element of two middle values of the list when the list has an even number of  
10 elements, or the median is a middle value element of the list when the list has an  
11 odd number of elements;

12 creating left side descendent nodes of the binary tree by successively  
13 finding a median of each left side grouping and linking each found median to the  
14 previous median, wherein a median of a first left side grouping is linked to the  
15 parent node;

16 creating right side descendent nodes of the binary tree by successively  
17 finding a median of each right side grouping and linking each found median to the  
18 previous median, wherein a median of a first right side grouping is linked to the  
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a  
21 left element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the  
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include data  
25 representing a number of times one or more threads of execution have passed  
through one or more code modules.

1 Claim 9 (currently amended): A method for creating a binary tree data  
2 structure, the data structure embodied in a computer-readable medium, from an  
3 ordered list of at least four elements, each element having an associated value in  
the list, comprising:

4 separating the list into left side groupings and right side groupings based on  
5 whether the list has an even or odd number of elements, the groupings being  
6 separated by a parent node defined by a median of the list, wherein the median is a  
7 left element of two middle values of the list when the list has an even number of  
8 elements, or the median is a middle value element of the list when the list has an  
odd number of elements;

9 creating left side descendent nodes of the binary tree by successively  
10 finding a median of each left side grouping and linking each found median to the  
11 previous median, wherein a median of a first left side grouping is linked to the  
12 parent node;

13 creating right side descendent nodes of the binary tree by successively  
14 finding a median of each right side grouping and linking each found median to the  
15 previous median, wherein a median of a first right side grouping is linked to the  
parent node;

16 wherein when a grouping has an even number of elements, the median is a  
17 left element of two middle values of the grouping;

18 wherein when a grouping has an odd number of number of elements, the  
19 median is a middle value element of the grouping; and

20 wherein the created right and left descendant nodes include one or more  
21 pointers to data representing a number of times one or more threads of execution  
22 have passed through one or more code modules.

23 Claim 10 (original): The method of claim 1, wherein the list is an ordered  
24 linked list.  
25

1 Claim 11 (canceled).

2 Claim 12 (canceled).

3  
4 Claim 13 (canceled).

5  
6 Claim 14 (canceled).

7  
8 Claim 15 (currently amended): A method for creating a binary tree data  
9 structure, the data structure embodied in a computer-readable medium, from an  
10 ordered list of at least four elements, each element having an associated value in  
the list, comprising:

11 determining whether the list has an even or odd number of elements;

12 separating the list into left side groupings and right side groupings based on  
13 whether the list has an even or odd number of elements, the groupings being  
14 separated by a parent node defined by a median of the list, wherein the median is a  
15 left element of two middle values of the list when the list has an even number of  
16 elements, or the median is a middle value element of the list when the list has an  
odd number of elements;

17 creating right side descendent nodes of the binary tree by successively  
18 finding a median of each right side grouping and linking each found median to the  
19 previous median, wherein a median of a first left side grouping is linked to the  
parent node;

20 creating left side descendent nodes of the binary tree by successively  
21 finding a median of each left side grouping and linking each found median to the  
22 previous median, wherein a median of a first left side grouping is linked to the  
23 parent node;

24 wherein when a grouping has an even number of elements, the median is a  
25 right element of two middle values of the grouping;

1 wherein when a grouping has an odd number of number of elements, the  
2 median is a middle value element of the grouping; and  
3 wherein the elements of the list include logged events.

4 Claim 16 (original): A computer-readable medium having stored thereon  
5 computer-executable instructions for performing the method of claim 15.

6  
7 Claim 17 (previously amended): The method of claim 15, wherein each  
8 element in the list includes a pointer to a corresponding node of a plurality of  
9 nodes in a partially assembled binary tree, wherein each node has a right child  
10 pointer, and wherein creating the right side nodes further comprises assigning a  
11 value to the right child pointer of at least one of the nodes.

12 Claim 18 (previously amended): The method of claim 15, wherein each  
13 element in the list includes a pointer to a corresponding node of a plurality of  
14 nodes in a partially assembled binary tree, wherein each node has a left child  
15 pointer, and wherein creating the left side nodes further comprises assigning a  
16 value to the left child pointer of at least one of the nodes.

17 Claim 19 (currently amended): The method of claim 15, wherein creating  
18 the right side descendent nodes comprises ~~creating~~ linking the right side  
19 descendent nodes ~~into to~~ a partially assembled version of the binary tree, wherein  
20 creating the left side descendent nodes comprises ~~creating~~ linking the left side  
21 descendent nodes ~~into to~~ the partially assembled version of the binary tree, and  
22 wherein the list is a linked list that acts as a wrapper around the partially  
23 assembled version of the binary tree.

24 Claim 20 (canceled).  
25

1 Claim 21 (currently amended): A method for creating a binary tree data  
2 structure, the data structure embodied in a computer-readable medium, from an  
3 ordered list of at least four elements, each element having an associated value in  
the list, comprising:

4 determining whether the list has an even or odd number of elements;  
5 separating the list into left side groupings and right side groupings based on  
6 whether the list has an even or odd number of elements, the groupings being  
7 separated by a parent node defined by a median of the list, wherein the median is a  
8 left element of two middle values of the list when the list has an even number of  
9 elements, or the median is a middle value element of the list when the list has an  
odd number of elements;

10 creating right side descendent nodes of the binary tree by successively  
11 finding a median of each right side grouping and linking each found median to the  
12 previous median, wherein a median of a first left side grouping is linked to the  
13 parent node;

14 creating left side descendent nodes of the binary tree by successively  
15 finding a median of each left side grouping and linking each found median to the  
16 previous median, wherein a median of a first right side grouping is linked to the  
parent node;

17 wherein when a grouping has an even number of elements, the median is a  
18 right element of two middle values of the grouping;

19 wherein when a grouping has an odd number of number of elements, the  
20 median is a middle value element of the grouping; and

21 wherein the elements of the list include data representing a number of times  
22 one or more threads of execution have passed through one or more code modules.  
23  
24  
25



1 Claim 22 (currently amended): A method for creating a binary tree data  
2 structure, the data structure embodied in a computer-readable medium, from an  
3 ordered list of at least four elements, each element having an associated value in  
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;  
6 separating the list into left side groupings and right side groupings based on  
7 whether the list has an even or odd number of elements, the groupings being  
8 separated by a parent node defined by a median of the list, wherein the median is a  
9 left element of two middle values of the list when the list has an even number of  
10 elements, or the median is a middle value element of the list when the list has an  
11 odd number of elements;

12 creating right side descendent nodes of the binary tree by successively  
13 finding a median of each right side grouping and linking each found median to the  
14 previous median, wherein a median of a first left side grouping is linked to the  
15 parent node;

16 creating left side descendent nodes of the binary tree by successively  
17 finding a median of each left side grouping and linking each found median to the  
18 previous median, wherein a median of a first right side grouping is linked to the  
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a  
21 right element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the  
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include data  
25 representing a number of times one or more threads of execution have passed  
through one or more code modules.

1 Claim 23 (currently amended): A method for creating a binary tree data  
2 structure, the data structure embodied in a computer-readable medium, from an  
3 ordered list of at least four elements, each element having an associated value in  
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;  
6 separating the list into left side groupings and right side groupings based on  
7 whether the list has an even or odd number of elements, the groupings being  
8 separated by a parent node defined by a median of the list, wherein the median is a  
9 left element of two middle values of the list when the list has an even number of  
10 elements, or the median is a middle value element of the list when the list has an  
11 odd number of elements;

12 creating right side descendent nodes of the binary tree by successively  
13 finding a median of each right side grouping and linking each found median to the  
14 previous median, wherein a median of a first left side grouping is linked to the  
15 parent node;

16 creating left side descendent nodes of the binary tree by successively  
17 finding a median of each left side grouping and linking each found median to the  
18 previous median, wherein a median of a first right side grouping is linked to the  
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a  
21 right element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the  
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include one or more  
25 pointers to data representing a number of times one or more threads of execution  
have passed through one or more code modules.

Claim 24 (original): The method of claim 15, wherein the list is an ordered  
linked list.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25

Claim 25 (canceled).

Claim 26 (canceled).

Claim 27 (canceled).

Claim 28 (canceled).

Claim 29 (previously amended): The method of claim 7, wherein each element in the list includes a pointer to a corresponding node of a plurality of nodes in a partially assembled binary tree, wherein each node has a left child pointer, and wherein creating the left side nodes further comprises assigning a value to the left child pointer of at least one of the nodes.

Claim 30 (previously amended): The method of claim 7 wherein each element in the list includes a pointer to a corresponding node of a plurality of nodes in a partially assembled binary tree, wherein each node has a right child pointer, and wherein creating the right side nodes further comprises assigning a value to the right child pointer of at least one of the nodes.

Claim 31 (currently amended): The method of claim 7, wherein creating the left side descendent nodes comprises ~~creating-linking~~ the left side descendent nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the right side descendent nodes comprises ~~creating-linking~~ the right side descendent nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the list is a linked list that acts as a wrapper around the partially assembled version of the binary tree.

1 Claim 32 (previously presented): The method of claim 7, wherein the list is  
2 an ordered linked list.

3 Claim 33 (previously presented): A computer-readable medium having  
4 stored thereon computer-executable instructions for performing the method of  
5 claim 7.

6  
7 Claim 34 (previously amended): The method of claim 8, wherein each  
8 element in the list includes a pointer to a corresponding node of a plurality of  
9 nodes in a partially assembled binary tree, wherein each node has a left child  
10 pointer, and wherein creating the left side nodes further comprises assigning a  
11 value to the left child pointer of at least one of the nodes.

12 Claim 35 (previously amended): The method of claim 8, wherein each  
13 element in the list includes a pointer to a corresponding node of a plurality of  
14 nodes in a partially assembled binary tree, wherein each node has a right child  
15 pointer, and wherein creating the right side nodes further comprises assigning a  
16 value to the right child pointer of at least one of the nodes.

17 Claim 36 (currently amended): The method of claim 8, wherein creating  
18 left side descendent nodes comprises ~~inserting~~ linking the left side descendent  
19 nodes ~~into to~~ a partially assembled version of the binary tree, wherein creating the  
20 right side descendent nodes comprises ~~creating~~ linking the right side descendent  
21 nodes ~~into to~~ the partially assembled version of the binary tree, and wherein the  
22 list is a linked list that acts as a wrapper around the partially assembled version of  
23 the binary tree.

24 Claim 37 (previously presented): The method of claim 8, wherein the list is  
25 an ordered linked list.

1           Claim 38 (previously presented): A computer-readable medium having  
2 stored thereon computer-executable instructions for performing the method of  
3 claim 8.

4  
5           Claim 39 (previously amended): The method of claim 9, wherein each  
6 element in the list includes a pointer to a corresponding node of a plurality of  
7 nodes in a partially assembled binary tree, wherein each node has a left child  
8 pointer, and wherein creating the left side nodes further comprises assigning a  
9 value to the left child pointer of at least one of the nodes.

10          Claim 40 (previously amended): The method of claim 9, wherein each  
11 element in the list includes a pointer to a corresponding node of a plurality of  
12 nodes in a partially assembled binary tree, wherein each node has a right child  
13 pointer, and wherein creating the right side nodes further comprises assigning a  
14 value to the right child pointer of at least one of the nodes.

15          Claim 41 (currently amended): The method of claim 9, wherein creating  
16 the left side descendent nodes comprises ~~creating-linking~~ the left side descendent  
17 nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the  
18 right side descendent nodes comprises ~~creating-linking~~ the right side descendent  
19 nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the  
20 list is a linked list that acts as a wrapper around the partially assembled version of  
21 the binary tree.

22          Claim 42 (previously presented): The method of claim 9, wherein the list is  
23 an ordered linked list.  
24  
25

1           Claim 43 (previously presented): A computer-readable medium having  
2 stored thereon computer-executable instructions for performing the method of  
3 claim 9.

4           Claim 44 (canceled).

5  
6           Claim 45 (canceled).

7           Claim 46 (canceled).

8  
9           Claim 47 (canceled).

10  
11          Claim 48 (previously amended): The method of claim 21, wherein each  
12 element in the list includes a pointer to a corresponding node of a plurality of  
13 nodes in a partially assembled binary tree, wherein each node has a right child  
14 pointer, and wherein creating the right side nodes further comprises assigning a  
15 value to the right child pointer of at least one of the nodes..

16          Claim 49 (previously amended): The method of claim 21, wherein each  
17 element in the list includes a pointer to a corresponding node of a plurality of  
18 nodes in a partially assembled binary tree, wherein each node has a left child  
19 pointer, and wherein creating the left side nodes further comprises assigning a  
20 value to the left child pointer of at least one of the nodes.

21          Claim 50 (currently amended): The method of claim 21, wherein creating  
22 the right side descendent nodes comprises ~~creating~~linking the right side  
23 descendent nodes ~~into~~to a partially assembled version of the binary tree, wherein  
24 creating the left side descendent nodes comprises ~~creating~~linking the left side  
25 descendent nodes ~~into~~to the partially assembled version of the binary tree, and

1 wherein the list is a linked list that acts as a wrapper around the partially  
2 assembled version of the binary tree.

3 Claim 51 (previously presented): The method of claim 21, wherein the list  
4 is an ordered linked list.

5  
6 Claim 52 (previously presented): A computer-readable medium having  
7 stored thereon computer-executable instructions for performing the method of  
8 claim 21.

9 Claim 53 (previously amended): The method of claim 22, wherein each  
10 element in the list includes a pointer to a corresponding node of a plurality of  
11 nodes in a partially assembled binary tree, wherein each node has a right child  
12 pointer, and wherein creating the right side nodes further comprises assigning a  
13 value to the right child pointer of at least one of the nodes.

14 Claim 54 (previously amended): The method of claim 22, wherein each  
15 element in the list includes a pointer to a corresponding node of a plurality of  
16 nodes in a partially assembled binary tree, wherein each node has a left child  
17 pointer, and wherein creating the left side nodes further comprises assigning a  
18 value to the left child pointer of at least one of the nodes.

19 Claim 55 (currently amended): The method of claim 22, wherein creating  
20 the right side descendent nodes comprises ~~creating~~ linking the right side  
21 descendent nodes ~~into~~ to a partially assembled version of the binary tree, wherein  
22 creating the left side descendent nodes comprises ~~creating~~ linking the left side  
23 descendent nodes ~~into~~ to the partially assembled version of the binary tree, and  
24 wherein the list is a linked list that acts as a wrapper around the partially  
25 assembled version of the binary tree.

1           Claim 56 (previously presented): The method of claim 22, wherein the list  
2 is an ordered linked list.

3  
4           Claim 57 (previously presented): The computer-readable medium having  
5 stored thereon computer-executable instructions for performing the method of  
6 claim 22.

7  
8           Claim 58 (previously amended): The method of claim 23, wherein each  
9 element in the list includes a pointer to a corresponding node of a plurality of  
10 nodes in a partially assembled binary tree, wherein each node has a right child  
11 pointer, and wherein creating the right side nodes further comprises assigning a  
12 value to the right child pointer of at least one of the nodes.

13           Claim 59 (previously amended): The method of claim 23, wherein each  
14 element in the list includes a pointer to a corresponding node of a plurality of  
15 nodes in a partially assembled binary tree, wherein each node has a left child  
16 pointer, and wherein creating the left side nodes further comprises assigning a  
17 value to the left child pointer of at least one of the nodes.

18           Claim 60 (currently amended): The method of claim 23, wherein creating  
19 the right side descendent nodes comprises ~~creating~~ linking the right side  
20 descendent nodes ~~into to~~ a partially assembled version of the binary tree, wherein  
21 creating the left side descendent nodes comprises ~~creating~~ linking the left side  
22 descendent nodes ~~into to~~ the partially assembled version of the binary tree, and  
23 wherein the list is a linked list that acts as a wrapper around the partially  
24 assembled version of the binary tree.  
25



1           Claim 61 (previously presented): The method of claim 23, wherein the list  
2 is an ordered linked list.

3           Claim 62 (previously presented): A computer-readable medium having  
4 stored thereon computer-executable instructions for performing the method of  
5 claim 23.

6           Claim 63 (currently amended): A method for creating a binary tree data  
7 structure, the data structure embodied in a computer-readable medium, from an  
8 ordered list of at least four elements, each element having an associated value in  
9 the list, comprising:

10           (a) determining whether the list has an even or odd number of elements;

11           (b) designating a median element of the list as a parent element based on  
12 whether the list has an even or odd number of elements, wherein the parent  
13 element divides the list into left side groupings and right side groupings, wherein  
14 the median is a right element of two middle values of the list when the list has an  
15 even number of elements, or the median is a middle value element of the list when  
the list has an odd number of elements;

16           (c) successively subdividing the right side groupings of the list and linking  
17 each successive median element with a previous median element, thereby creating  
18 right side descendent nodes in the binary tree, and wherein a median of a first right  
19 side grouping is linked to the parent element;

20           (d) once each right side grouping has been exhausted as a result of step (c),  
21 stepping back up the tree through each successive ancestor node until reaching an  
22 element having left side groupings in the list, and, upon reaching an element  
having a left side grouping in the list, proceeding to step (e);

23           (e) subdividing the left side groupings and linking a median element of ~~the~~  
24 a left side grouping with the element reached in step (d), thereby creating a left  
25 side descendent of the binary tree;

1 (f) if the left side descendent of step (e) has a right side grouping in the list,  
2 repeating step (c) for the right side grouping;

3 (g) if the left side descendent of step (e) has no right side groupings, but has  
4 a left side grouping, repeating step (e) for the left side grouping;

5 wherein the median element of a grouping is a right element of two middle  
6 values of the grouping when the grouping has an even number of elements, or the  
7 median is a middle value element of the grouping when the list has an odd number  
8 of number of elements; and

9 wherein the elements of the list include data representing number of times  
10 one or more threads of execution have passed through one or more code modules.

11 Claim 64 (previously presented): A computer-readable medium having  
12 stored thereon computer-executable instructions for performing the method of  
13 claim 63.

14 Claim 65 (previously amended): A method for creating a binary tree data  
15 structure, the data structure embodied in a computer-readable medium, from an  
16 ordered list of at least four elements, each element having an associated value in  
17 the list, comprising:

18 (a) determining whether the list has an even or odd number of elements;

19 (b) designating a median element of the list as a parent element based on  
20 whether the list has an even or odd number of elements, wherein the parent  
21 element divides the list into left side groupings and right side groupings, wherein  
22 the median is a right element of two middle values of the list when the list has an  
23 even number of elements, or the median is a middle value element of the list when  
24 the list has an odd number of elements;

25 (c) determining if there are elements to the right of the parent element;

(d) if there are no elements to the right of the parent element, proceeding to  
step (h);

1 (e) for the elements that are to the right of the parent element, finding a  
median element;

2 (f) linking the median element of step (e) to the parent element so that the  
3 median element is a child of the parent element;

4 (g) repeating steps (d) and (e), wherein the child element of step (f) is now  
5 treated as the parent element in steps (d) and (e);

6 (h) locating a next element up on the tree that has elements to the left of it  
7 and treating the element as a parent element in step (i);

8 (i) finding a median element of the elements to the left of the parent  
9 element from step (h);

10 (j) linking the median element of step (i) to the parent element of step (h),  
wherein the median element is a child of the parent;

11 (k) repeating steps (d) through (g), wherein the child element of step (j) is  
12 treated as the parent element in step (d);

13 wherein the median element of a grouping is a right element of two middle  
14 values of the grouping when the grouping has an even number of elements, or the  
15 median is a middle value element of the grouping when the list has an odd number  
of number of elements; and

16 wherein the elements of the list include data representing number of times  
17 one or more threads of execution have passed through one or more code modules.

18  
19 Claim 66 (previously presented): A computer-readable medium having  
20 stored thereon computer-executable instructions for performing the method of  
21 claim 65.  
22  
23  
24  
25